IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Ramachandran et al.**	§	Confirmation No.: **9668**
	§	
Serial No.: **10/626,205**	§	Group Art Unit: **2192**
	§	
Filed: **July 24, 2003**	§	Examiner: **Pham, Chrystine**
	§	
For: **Method and Apparatus for Monitoring Compatibility of Software Combinations**	§	Attorney Docket No.: **AUS920030501US1**

Commissioner for Patents
P.O. Box 1450
Alexandria VA 22313-1450

35525
PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

RESPONSE TO OFFICE ACTION

Sir:

No fees are believed to be required. If, however, any fees are required, I authorize the Commissioner to charge these fees, which may be required to IBM Corporation Deposit Account No. 09-0447. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to IBM Corporation Deposit Account No. 09-0447.

In response to the Office Action of November 23, 2007, please amend the above-identified application as follows:

Amendments to the Claims are reflected in the listing of claims, which begins on page 2 of this paper.

Remarks/Arguments begin on page 5 of this paper.

Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for testing the compatibility of software modules, the method comprising the computer implemented steps of:

responsive to receiving a request to install a new software module in a data processing system, performing an inventory on an existing set of software modules resident in the data processing system;

referring to a knowledge base of software modules to determine whether the new software module is known to function compatibly with the existing set of software modules; ~~to form compatibility information; and~~

responsive to a determination that the new software module is not known to function compatibly with the existing set of software modules, referring to the knowledge base of software modules to determine whether the new software module is known to function incompatibly with the existing set of software modules; and

responsive to a determination that the new software module is not known to function incompatibly with the existing set of software modules, testing the new software module in a test data processing system in combination with the existing set of software modules.

~~providing the compatibility information, wherein the compatibility information is used to determine whether to install the new software module.~~

2. (Currently Amended) The method of claim 1, ~~further comprising:~~ wherein testing the new software module in a test data processing system in combination with the existing set of software modules comprises:

prompting a user for a first selected input;

responsive to a first selected user input, testing the new software module in a test data processing system in combination with the existing set of software modules; and

[[and]] responsive to a second selected user input, installing the new software module in the data processing system.

3. (Currently Amended) The method of claim 1, further comprising:

responsive to a determination that the new software module is known to function incompatibly with the existing set of software modules, prompting a user for a first input;

responsive to a first selected user input, testing the new software module in a test data processing system in combination with the existing set of software modules; and

responsive to a second selected user input, installing the new software module in the data processing system.

~~A method for testing the compatibility of software modules, the method comprising the computer implemented steps of:~~

~~responsive to receiving a request to install a new software module in a data processing system, performing an inventory on an existing set of software modules resident in the data processing system;~~

~~referring to a knowledge base of software modules to determine whether the new software module is known to function compatibly with the existing set of software modules; and~~

~~responsive to a negative determination, testing the new software module in a test data processing system in combination with the existing set of software modules.~~

4. (Currently Amended) The method of claim 1, [[3]] further comprising:

responsive to ~~a determination~~ a test result indicating that the new software module is compatible with the existing software modules, adding a new combination indicating the compatibility to the knowledge base; and

installing the new software module in the data processing system.

5. (Currently Amended) The method of claim 1, [[3]] further comprising:

responsive to a determination that the new software module is not compatible with the existing software modules, adding a new combination indicating the incompatibility to the knowledge base and searching the knowledge base to find a closest match, wherein at least one of the existing modules is removed or replaced with a different version;

prompting for the user as to availability of the closest match combination; and

responsive to a user input, installing the new software module and changing the existing modules as needed to obtain a compatible combination.

6. (Cancelled)

7. (Currently Amended) The method of claim 1, [[6,]] wherein ~~the installing step~~ testing the new software module in a test data processing system in combination with the existing set of software modules comprises:

identifying an environment of a client in which the software module is to be installed;

recreating the environment on a test data processing system; and

installing the software module on the test data processing system to form the installed software module.

8.-19. (Cancelled)

REMARKS/ARGUMENTS

Claims 1-19 are pending in the present application. Claims 6 and 8-19 were cancelled; claims 1-5 and 7 were amended; and no claims were added. No new matter has been added by any of the amendments to the claims. Support for the amendments to the claims may be found in the specification on at least page 11, lines 20-22; page 14, lines 18-22; page 15, line 20 – page 16, line 10; page 16, lines 23-29; page 17, lines 1-8; page 18, line 11 – page 19, line 2; Figure 4, 402; Figure 5, 512; Figure 6 610, 622; Figure 7 704-708. Reconsideration of the claims is respectfully requested.

Applicants have amended claims 1-5 and 7 and cancelled claims 6 and 8-19 from further consideration in this application. Applicants are not conceding in this application that those claims are not patentable over the art cited by the Examiner, as the present claim amendments and cancellations are only for facilitating expeditious prosecution of the present application. Applicants respectfully reserve the right to pursue these and other claims in one or more continuations and/or divisional patent applications.

I. 35 U.S.C. § 102, Anticipation

The Office Action has rejected claims 1-6, 8-11, 13-16, and 18-19 under 35 U.S.C. § 102 as being anticipated by *Kroening et al.*, Apparatus and Method for Checking Component Compatibility in a Build to Order Computer System, U.S. Patent No. 6,735,757, May 11, 2004 (hereinafter "*Kroening*"). This rejection is respectfully traversed.

Regarding this rejection, the Office Action states:

> Claim 1
>
> Kroening teaches a method for testing the compatibility of software modules (see at least *checking component compatibility, list of components, version/revision number, compatibility property* co1.2:24-36), the method comprising the computer-implemented steps of:
>
> responsive to receiving a request to install a new software module in a data processing system (see at least 405 FIG.4 & associated text), performing an inventory (see at least 120, 130 FIG. 1 & associated text) on an existing 'set of software modules resident in the data processing system (see at least 425, 430 FIG.4 & associated text; 210, 230, 240 FIG.2 & associated text; *component 210, component table, database, identification property 220, compatibility property 240, version/revision property 250* co1.3: 13-38);
>
> referring to a knowledge base of software modules to determine whether the new software module is known to function compatibly with the existing set of software modules to form compatibility information (see at least *conflicts, component, no compatibility issues, installed 450* co1.4: 18-37); and

providing the compatibility information, wherein the compatibility information is used to determine whether to install the new software module (see at least *conflicts, component,' no compatibility issues, installed 450* co1.4: 18-37; 435, 450 FIG. 4 & associated text).

Office Action dated November 23, 2007, pp. 3-4 (emphasis in original).

A prior art reference anticipates the claimed invention under 35 U.S.C. § 102 only if every element of a claimed invention is identically shown in that single reference, arranged as they are in the claims. *In re Bond*, 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990). All limitations of the claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). Anticipation focuses on whether a claim reads on the product or process a prior art reference discloses, not on what the reference broadly teaches. *Kalman v. Kimberly-Clark Corp.*, 713 F.2d 760, 218 U.S.P.Q. 781 (Fed. Cir. 1983). In this case, each and every feature of the presently claimed invention is not identically shown in the cited reference, arranged as they are in the claims.

Amended independent claim 1 recites:

1. (Previously Presented) A method for testing the compatibility of software modules, the method comprising the computer implemented steps of:

responsive to receiving a request to install a new software module in a data processing system, performing an inventory on an existing set of software modules resident in the data processing system;

referring to a knowledge base of software modules to determine whether the new software module is known to function compatibly with the existing set of software modules;

responsive to a determination that the new software module is not known to function compatibly with the existing set of software modules, referring to the knowledge base of software modules to determine whether the new software module is known to function incompatibly with the existing set of software modules; and

responsive to a determination that the new software module is not known to function incompatibly with the existing set of software modules, testing the new software module in a test data processing system in combination with the existing set of software modules.

Kroening fails to anticipate amended claim 1, as *Kroening* fails to teach each and every feature of amended claim 1. Specifically, *Kroening* fails to teach the features of "referring to a knowledge base of software modules to determine whether the new software module is known to function compatibly with the existing set of software modules," "responsive to a determination that the new software module is not known to function compatibly with the existing set of software modules, referring to the knowledge base of software modules to determine whether the new software module is known to function incompatibly with the existing set of software modules" and "responsive to a determination that the new software module is not known to function incompatibly with the existing set of software modules, testing the new software module in a test data processing system in combination with the existing set of software

modules." The Office Action cites to *Kroening*, column 4, lines 18-37, which is reproduced below for the Examiner's convenience, as teaching referring to a knowledge base of software modules to determine whether the new software module is known to function compatibly with the existing set of software modules to form compatibility information:

> At this time, each installation and testing step **420** may be analyzed to determine the identity of the component **425** and the version/revision number **430**. The apparatus of the present invention checks if there are any known conflicts present in the component involved in the associated step **435**. If there are no compatibility issues with regard to the component, then the component is installed **450** and the process moves to the next step **455**. If all the testing and installation steps have been completed, then the process is complete **460**.
>
> If the component has a compatibility conflict, then the apparatus of the present invention checks if any non-conflicting versions/revisions of the component are available **440**. In a preferred embodiment, the apparatus may utilize the latest version/revision of the component that does not have compatibility conflicts. If another version/revision of the component is available that does not have a compatibility conflict, then that particular version/revision of the component may be installed.

Kroening, column 4, lines 18-37 (emphasis in original).

The above cited passage of *Kroening* fails to teach the features of "responsive to a determination that the new software module is not known to function compatibly with the existing set of software modules, referring to the knowledge base of software modules to determine whether the new software module is known to function incompatibly with the existing set of software modules" and "responsive to a determination that the new software module is not known to function incompatibly with the existing set of software modules, testing the new software module in a test data processing system in combination with the existing set of software modules." Rather, as taught by *Kroening*, each component involved in a particular installation sequence is checked to see if there are **known conflicts with the other components involved in the associated installation step**. If no compatibility issues are found to exist, then the component is installed. If compatibility issues do exist, then *Kroening* teaches searching for non-conflicting versions of the component and installing that component, if available.

Kroening makes the determination about compatibility issues by checking a compatibility property **240** of each component involved in that particular installation step. *Kroening*, in column 3, lines 58 through 60, states that "the compatibility defines known compatibility issues between components." Thus, *Kroening* only teaches installing software if there are non-previously known compatibility issues with other software involved in that particular installation step. Further, in column 3, line 64, though column 4, line 1, *Kroening* teaches that software is only tested when a software or hardware component that is listed as having a known compatibility issue with a previously installed software module is upgraded to a new version. Then, the new, upgraded version is tested to see if the prior compatibility issues still exist.

Thus, the invention taught by *Kroening* is very different from the invention recited in claim 1 for a number of reasons. First, *Kroening* teaches a single pronged test to decide whether to install a component. The test comprises determining, based on a compatibility property that lists known compatibility issues, whether the component has any known compatibility issues with other components being installed during that installation step. If there are no known compatibility issues with other comments being installed during that step, then the component is installed.

In contradistinction, claim 1 recites a two-pronged test to determine whether to test the component. Claim 1 recites, "responsive to a determination that the new software module is not known to function compatibly with the existing set of software modules, referring to the knowledge base of software modules to determine whether the new software module is known to function incompatibly with the existing set of software modules" and "responsive to a determination that the new software module is not known to function incompatibly with the existing set of software modules, testing the new software module in a test data processing system in combination with the existing set of software modules." Thus, as recited in claim 1, a knowledge base is checked to see if there is information present that indicates that the new component and the prior installed components function compatibly. If there is not information indicating that the new component and the prior installed components function compatibly, the knowledge base is checked to see if there is information present that indicates that the new component and the prior installed components function incompatibly. If there is not information indicating that the new component and the prior installed components function incompatibly, then the new software is tested in a test data processing system.

Kroening does not teach determining "whether the new software module is known to function compatibly with the existing set of software modules," as recited in claim 1. Rather, *Kroening* teaches determining whether any compatibility issues are known.

Kroening also does not teach "responsive to a determination that the new software module is not known to function compatibly with the existing set of software modules, referring to the knowledge base of software modules to determine whether the new software module is known to function incompatibly with the existing set of software modules." Instead, *Kroening* teaches that if there are not any known compatibility issues, the component simply is installed. Therefore, as taught by *Kroening*, components that do have compatibility issues could still be installed if those compatibility issues were not previously known, thus causing problems after installation. In contradistinction, as recited by claim 1, any component whose compatibility with existing software modules is not known, will be tested on a separate test data processing system. Therefore, potential conflicts are identified on the test data processing system before the component is installed, thus solving the problem present in the invention of *Kroening*.

Additionally, *Kroening* fails to teach the feature of "responsive to a determination that the new software module is not known to function incompatibly with the existing set of software modules, testing the new software module in a test data processing system in combination with the existing set of software modules." *Kroening* teaches testing a new version of software that is known to have compatibility issues to see if the compatibility issues have been resolved. Applicants respectfully submit that this is very different from the feature of "responsive to a determination that the new software module is not known to function incompatibly with the existing set of software modules, testing the new software module in a test data processing system in combination with the existing set of software modules," recited in claim 1.

Further *Kroening* teaches that compatibility issues are only checked for "components involved in the associated step **435**" of the installation process. (See *Kroening*, column 4, lines 20-22) In contradistinction, claim 1 recites determining "whether the new software module is known to function compatibly **with the existing set of software modules.**" However, *Kroening* only teaches determining compatibility issues with a limited subset of components that make up the data processing system.

Therefore, for at least the reasons set forth above, Applicants respectfully submit that *Kroening* fails to anticipate claim 1 as *Kroening* fails to teach each and every feature of claim 1. Thus, Applicants respectfully submit that claim 1 is in condition for allowance over the cited reference of *Kroening*. Since claims 2-5 depend from claim 1, the same distinctions between *Kroening* and the claimed invention in claim 1 applies for these claims. Consequently, Applicants respectfully submit that claims 2-5 are also in condition for allowance over the *Kroening* reference at least by virtue of their depending from an allowable base claim.

Additionally, claim 4 recites features not taught by the *Kroening* reference. Claim 4 recites the feature of "responsive to a test result indicating that the new software module is compatible with the existing software modules, adding a new combination indicating the compatibility to the knowledge base." *Kroening* fails to teach either of these features. As stated in column 3, lines 60-64 of *Kroening*, the compatibility property is only updated when new compatibility issues are found. However, claim 4 recites updating the knowledge base when software is found to have no compatibility issues with the existing software modules.

Claims 8-11, 13-16, and 18-19 have been cancelled without prejudice.

Therefore, the rejection of claims 1-6, 8-11, 13-16, and 18-19 under 35 U.S.C. § 102 has been overcome.

II. 35 U.S.C. § 103, Obviousness

The Office Action has rejected claims 7, 12, and 17 under 35 U.S.C. § 103 as being unpatentable over *Kroening* in view of *Lau,* Method and System for Testing Provisioning and Interoperability of Computer System Services, U.S. Patent Publication No. 2004/0128651, April 1, 2007 (hereinafter "*Lau*"). This rejection is respectfully traversed.

Regarding this rejection, the Office Action states:

> Claim 7
>
> The rejection of base claim 6 is incorporated. Kroening further teaches wherein the installing step comprises:
>
> identifying an environment of a client in which the software module is to be installed (see at least 405 FIG.4 & associated text);
>
> Kroening does not expressly disclose:
>
> recreating the environment on a test data processing system; and
>
> installing the software module on the test data processing system to form the installed software module.
>
> However, Lau discloses a method and system for software interoperability testing (see at least 118, 146 FIG. 1 & associated text; *interoperability tests, new software tool, existing system software* paragraph [0010]) comprising:
>
> identifying an environment of a client (see at least 140 FIG. 1 & associated text) in which the software module is to be installed (see at least 230 FIG.2 & associated text)
>
> recreating the environment on a test data processing system (see at least 116, 1 10 FIG. 1 & associated text); and
>
> installing the software module on the test data processing system to form the installed software module (see at least paragraphs [0015]; [0023]; [0027]; [0028]).
>
> Kroening and Lau are analogous art because they are both directed to software interoperability testing. It would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of Lau into that of Kroening for the inclusion of recreating the environment on a test data processing system; and installing the software module on the test data processing system to form the installed software module. And the motivation for doing so would have been to enable remote testing of software interoperability for multiple networked computer systems prior to performing software migration (see at least Lau paragraphs

Office Action dated November 23, 2007, pp. 10-11 (emphasis in original).

The Examiner bears the burden of establishing a *prima facie* case of obviousness based on prior art when rejecting claims under 35 U.S.C. § 103. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). The prior art reference (or references when combined) must teach or suggest all the claim limitations. *In re Royka,* 490 F.2d 981, 180 USPQ 580 (CCPA 1974). In determining obviousness, the scope and content of the prior art are… determined; differences between the prior art and the claims at issue are… ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background the obviousness or non-obviousness of the subject matter is determined. *Graham v. John Deere Co.*, 383 U.S. 1 (1966). "Often, it will be necessary for a court to look to interrelated teachings of multiple patents; the effects of demands known to the design community or present in the marketplace;

and the background knowledge possessed by a person having ordinary skill in the art, all in order to determine whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue." *KSR Int'l. Co. v. Teleflex, Inc.*, No. 04-1350 (U.S. Apr. 30, 2007). "*Rejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness.* Id. (citing *In re Kahn*, 441 F.3d 977, 988 (CA Fed. 2006))."

Amended claim 7 depends from amended claim 1 and claims 12 and 17 have been cancelled without prejudice. As argued above, *Kroening* fails to teach all the features of claim 1. *Lau* fails to cure the deficiencies of *Kroening* as *Lau* fails to teach the features missing from *Kroening*, the features of "referring to a knowledge base of software modules to determine whether the new software module is known to function compatibly with the existing set of software modules," "responsive to a determination that the new software module is not known to function compatibly with the existing set of software modules, referring to the knowledge base of software modules to determine whether the new software module is known to function incompatibly with the existing set of software modules" and "responsive to a determination that the new software module is not known to function incompatibly with the existing set of software modules, testing the new software module in a test data processing system in combination with the existing set of software modules." The Office Action does not cite to any portion of *Lau* as teaching these features, nor does any portion of *Lau* teach these features.

Lau is directed to a "method for testing software and hardware components or systems for service provisioning or interoperability with hardware and software of a computer system." (See *Lau*, Abstract) As shown in Figure 2 and the accompanying text in paragraph [0032] of *Lau*, the method of *Lau* begins by receiving a request to perform a test. Thus, the invention of *Lau* takes place after a determination to perform a test has already been made. As such, *Lau* is silent in regards to the features of "referring to a knowledge base of software modules to determine whether the new software module is known to function compatibly with the existing set of software modules," "responsive to a determination that the new software module is not known to function compatibly with the existing set of software modules, referring to the knowledge base of software modules to determine whether the new software module is known to function incompatibly with the existing set of software modules" and "responsive to a determination that the new software module is not known to function incompatibly with the existing set of software modules, testing the new software module in a test data processing system in combination with the existing set of software modules." Therefore, *Lau* does not teach, suggest or even hint at the features of "referring to a knowledge base of software modules to determine whether the new software module is known to function compatibly with the existing set of software modules," "responsive to a determination that the new software module is not known to function compatibly with the existing set of software

modules, referring to the knowledge base of software modules to determine whether the new software module is known to function incompatibly with the existing set of software modules" and "responsive to a determination that the new software module is not known to function incompatibly with the existing set of software modules, testing the new software module in a test data processing system in combination with the existing set of software modules." Thus, *Lau* fails to cure the deficiencies of *Kroening*.

Therefore, for at least the reasons set forth above, Applicants respectfully submit that *Kroening* and *Lau*, either alone or in combination, fail to teach or suggest all the features of claim 1. Thus, Applicants respectfully submit that claim 1 is in condition for allowance over the combination of *Kroening* in view of *Lau*. Since claim 7 depends from claim 1, the same distinctions between the cited combination of *Kroening* in view of *Lau* and the claimed invention in claim 1 applies for claim 7. Consequently, Applicants respectfully submit that claim 7 is also in condition for allowance over the combination of *Kroening* in view of *Lau* at least by virtue of depending from an allowable base claim.

Therefore, the rejection of claims 7, 12, and 17 under 35 U.S.C. § 103 has been overcome.

III. Conclusion

It is respectfully urged that the subject application is patentable in view of the cited references and is now in condition for allowance.

The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: February 20, 2008

Respectfully submitted,

/Gerald H. Glanzman/

Gerald H. Glanzman
Reg. No. 25,035
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorney for Applicants

GG/blj